数据结构

前置知识

二分，分治，倍增，线段树，树状数组，启发式合并，树链剖分，单调栈，单调队列，平衡树（FHQ，Splay 二选一），分块，莫队的模板与基本应用。

CDQ 分治

- 一般来说问题会有一个时间轴，每个时刻可能会有修改或查询操作。

CDQ 分治

- 一般来说问题会有一个时间轴，每个时刻可能会有修改或查询操作。
- 如果能将问题离线，且修改对查询的贡献独立，那么考虑一个分治结构，则每次只需要考虑一侧的修改对另一侧查询的贡献。

CDQ 分治

- 一般来说问题会有一个时间轴，每个时刻可能会有修改或查询操作。
- 如果能将问题离线，且修改对查询的贡献独立，那么考虑一个分治结构，则每次只需要考虑一侧的修改对另一侧查询的贡献。
- 如果处理贡献的复杂度和操作数量相关那么可以得到比较优的复杂度。

例题 · P3810 【模板】三维偏序

例题 · P4169 [Violet] 天使玩偶

线段树分治

- 同样存在一个时间轴，且每个元素会在时间轴上存在一段区间（也就是支持插入和删除操作）。

线段树分治

- 同样存在一个时间轴，且每个元素会在时间轴上存在一段区间（也就是支持插入和删除操作）。
- 如果只支持插入是容易的，我们能利用线段树分治将删除操作套路地转化为撤回操作，于是便无需再单独实现删除。

线段树分治

- 同样存在一个时间轴，且每个元素会在时间轴上存在一段区间（也就是支持插入和删除操作）。
- 如果只支持插入是容易的，我们能利用线段树分治将删除操作套路地转化为撤回操作，于是便无需再单独实现删除。
- 具体来说，我们离线后把时间轴建立线段树，随后我们把所有元素扔到对应的线段所覆盖的线段树上的区间上。

线段树分治

- 同样存在一个时间轴，且每个元素会在时间轴上存在一段区间（也就是支持插入和删除操作）。
- 如果只支持插入是容易的，我们能利用线段树分治将删除操作套路地转化为撤回操作，于是便无需再单独实现删除。
- 具体来说，我们离线后把时间轴建立线段树，随后我们把所有元素扔到对应的线段所覆盖的线段树上的区间上。
- 随后开始在线段树上 DFS，每到一个节点就将节点上的元素插入。撤回一个节点就撤销这次对应的插入。

线段树分治

- 同样存在一个时间轴，且每个元素会在时间轴上存在一段区间（也就是支持插入和删除操作）。
- 如果只支持插入是容易的，我们能利用线段树分治将删除操作套路地转化为撤回操作，于是便无需再单独实现删除。
- 具体来说，我们离线后把时间轴建立线段树，随后我们把所有元素扔到对应的线段所覆盖的线段树上的区间上。
- 随后开始在线段树上 DFS，每到一个节点就将节点上的元素插入。撤回一个节点就撤销这次对应的插入。
- 复杂度相当于在不带删除的情况下增加一个 log。

例题 · P5787 二分图

猫树分治

- 对于一类静态问题，通常维护合并的复杂度代价很大，而单独加入一个元素的复杂度代价相对可接受，则可尝试猫树分治。

猫树分治

- 对于一类静态问题，通常维护合并的复杂度代价很大，而单独加入一个元素的复杂度代价相对可接受，则可尝试猫树分治。
- 我们仍然建立出一棵分治树，每一层每一段维护左段的后缀信息和右段的前缀信息，则每次查询找到第一个跨过查询区间的段，只需要做一次左端与右端的合并即可。

猫树分治

- 对于一类静态问题，通常维护合并的复杂度代价很大，而单独加入一个元素的复杂度代价相对可接受，则可尝试猫树分治。
- 我们仍然建立出一棵分治树，每一层每一段维护左段的后缀信息和右段的前缀信息，则每次查询找到第一个跨过查询区间的段，只需要做一次左端与右端的合并即可。
- 这样相比直接用线段树维护，合并次数由 $\mathrm{O}(\log n)$ 变为了 $\mathrm{O}(1)$，便能在特定情况下优化复杂度。

例题 · P6240 好吃的题目

更多例题

- CDQ 分治
  - P3769 [CH 弱省胡策 R2] TATT
  - CF1045G AI robots
  - P4027 [NOI2007] 货币兑换
- 线段树分治
  - P4585 [FJOI2015] 火星商店问题
  - CF576E Painting Edges
  - CF603E Pastoral Oddities
- 猫树分治
  - CF1100F Ivan and Burgers

例题 · 忘了来源的题目 #1

给定一个单调不降函数 f 和一个单调不升函数 g，求最大的 $\min(f(x), g(x))$。要求做到 $\mathrm{O}(\log V)$。

整体二分

- 对于一类多次询问问题，且单次询问二分是容易解决的，但多次询问会由于不同 mid 有不同状态而复杂度过高，那么可以考虑整体二分。

整体二分

- 对于一类多次询问问题，且单次询问二分是容易解决的，但多次询问会由于不同 mid 有不同状态而复杂度过高，那么可以考虑整体二分。
- 考虑把所有询问离线后一起二分，设置一个指针 now 并维护当前 mid 为 now 的状态，将 now 调整至当前区间的 mid。

整体二分

- 对于一类多次询问问题，且单次询问二分是容易解决的，但多次询问会由于不同 mid 有不同状态而复杂度过高，那么可以考虑整体二分。
- 考虑把所有询问离线后一起二分，设置一个指针 now 并维护当前 mid 为 now 的状态，将 now 调整至当前区间的 mid。
- check 一遍当前区间内所有询问，于是可以根据大小关系将询问分为两个集合，并分别扔到左右子区间中去处理即可。

整体二分

- 对于一类多次询问问题，且单次询问二分是容易解决的，但多次询问会由于不同 mid 有不同状态而复杂度过高，那么可以考虑整体二分。
- 考虑把所有询问离线后一起二分，设置一个指针 now 并维护当前 mid 为 now 的状态，将 now 调整至当前区间的 mid。
- check 一遍当前区间内所有询问，于是可以根据大小关系将询问分为两个集合，并分别扔到左右子区间中去处理即可。
- 可以发现合理移动下 now 的移动次数为 $\mathrm{O}(n \log n)$，且每次询问会 check $\mathrm{O}(\log n)$ 次。

例题 · P1527 [国家集训队] 矩阵乘法

例题 · P3527 [POI2011] MET-Meteors

线段树的适用范围

- 支持查询和单点修改只需要满足：能有效地合并两段的信息。

线段树的适用范围

- 支持查询和单点修改只需要满足：能有效地合并两段的信息。
- 支持批量插入需要额外满足：设计一个标记，并能有效地将标记与标记合并以及标记与权值合并。

例题 · P6327 区间加区间 sin 和

扫描线

- 对于一类二维，三维甚至多维问题，我们可以考虑枚举其中一维，用数据结构维护另一维。

扫描线

- 对于一类二维，三维甚至多维问题，我们可以考虑枚举其中一维，用数据结构维护另一维。
- 如果每次移动其中一维对另一维的影响能描述为数据结构容易解决的问题那么就可以用扫描线解决。

扫描线

- 对于一类二维，三维甚至多维问题，我们可以考虑枚举其中一维，用数据结构维护另一维。
- 如果每次移动其中一维对另一维的影响能描述为数据结构容易解决的问题那么就可以用扫描线解决。
- 枚举的一维可以是 y 轴，时间轴，右端点轴等，比较灵活。经典例子二维数点。

例题 · P5490 【模板】扫描线

例题 · GYM102222L Continuous Intervals

可持久化

- 一般来说我们的线段树会进行许多次操作，经历许多状态，但朴素的线段树只允许我们查询当前的状态。

可持久化

- 一般来说我们的线段树会进行许多次操作，经历许多状态，但朴素的线段树只允许我们查询当前的状态。
- 就好像用扫描线解决区间问题，如果问题必须强制在线那么我们就必须能查询任意历史状态下的线段树。

可持久化

- 一般来说我们的线段树会进行许多次操作，经历许多状态，但朴素的线段树只允许我们查询当前的状态。
- 就好像用扫描线解决区间问题，如果问题必须强制在线那么我们就必须能查询任意历史状态下的线段树。
- 可持久化正是记录了所有的历史版本，但是为了高效我们每个版本只记录相比上个版本有变化的节点，其余不变点则直接继承，于是时间复杂度不会变劣，而空间复杂度会显著增加。

线段树合并

- 首先需要了解动态开点线段树，即当线段树上大部分节点没有意义时，我们只需要保留那些有意义的节点。

线段树合并

- 首先需要了解动态开点线段树，即当线段树上大部分节点没有意义时，我们只需要保留那些有意义的节点。
- 这时有若干棵不完整的线段树，我们定义了一种同态节点的合并方式并想将这些线段树合并，则可以考虑以两棵两棵的顺序合并。

线段树合并

- 首先需要了解动态开点线段树，即当线段树上大部分节点没有意义时，我们只需要保留那些有意义的节点。
- 这时有若干棵不完整的线段树，我们定义了一种同态节点的合并方式并想将这些线段树合并，则可以考虑以两棵两棵的顺序合并。
- 对于一次合并，我们其实可以直接暴力解决，即两棵树上同时有信息的点就递归儿子继续做，否则就只继承有信息的那棵树的信息并返回即可。

线段树合并

- 首先需要了解动态开点线段树，即当线段树上大部分节点没有意义时，我们只需要保留那些有意义的节点。
- 这时有若干棵不完整的线段树，我们定义了一种同态节点的合并方式并想将这些线段树合并，则可以考虑以两棵两棵的顺序合并。
- 对于一次合并，我们其实可以直接暴力解决，即两棵树上同时有信息的点就递归儿子继续做，否则就只继承有信息的那棵树的信息并返回即可。
- 由于每次递归会导致所有线段树的总节点数减少，所以时空复杂度是与所有线段树的节点总数线性相关的。同时合并的过程也能可持久化，但这会导致空间复杂度常数变大。

更多例题

- SP1716 GSS3 - Can you answer these queries III
- P2572 [SCOI2010] 序列操作
- P2163 [SHOI2007] 园丁的烦恼
- CF407E k-d-sequence

李超树

- 二维平面上有许多直线（线段），我们需要支持插入一个直线（线段）以及查询给定 x 处所有线段的最大值（最小值）是什么。

李超树

- 二维平面上有许多直线（线段），我们需要支持插入一个直线（线段）以及查询给定 x 处所有线段的最大值（最小值）是什么。
- 考虑用动态开点线段树维护 x 轴，每个节点上存储在 mid 处最大的那条线段，于是每插入一条直线可以分类讨论是否与当前 mid 处的线段交换，而在 mid 处较小的那个线段一定在左右两侧中的一侧都不如 mid 处较大的线段。

李超树

- 二维平面上有许多直线（线段），我们需要支持插入一个直线（线段）以及查询给定 x 处所有线段的最大值（最小值）是什么。
- 考虑用动态开点线段树维护 x 轴，每个节点上存储在 mid 处最大的那条线段，于是每插入一条直线可以分类讨论是否与当前 mid 处的线段交换，而在 mid 处较小的那个线段一定在左右两侧中的一侧都不如 mid 处较大的线段。
- 于是我们每次只会向一侧递归，同时可以发现询问时只需要把路径上的每个线段都取个 max 即可，于是可以做到 $\mathrm{O}(n \log V)$。

李超树

- 二维平面上有许多直线（线段），我们需要支持插入一个直线（线段）以及查询给定 x 处所有线段的最大值（最小值）是什么。
- 考虑用动态开点线段树维护 x 轴，每个节点上存储在 mid 处最大的那条线段，于是每插入一条直线可以分类讨论是否与当前 mid 处的线段交换，而在 mid 处较小的那个线段一定在左右两侧中的一侧都不如 mid 处较大的线段。
- 于是我们每次只会向一侧递归，同时可以发现询问时只需要把路径上的每个线段都取个 max 即可，于是可以做到 $\mathrm{O}(n \log V)$。
- 当插入的不是直线而是线段时我们需要先分割为若干区间再单独插入，此时复杂度为 $\mathrm{O}(n \log^2 V)$。

李超树

- 二维平面上有许多直线（线段），我们需要支持插入一个直线（线段）以及查询给定 x 处所有线段的最大值（最小值）是什么。
- 考虑用动态开点线段树维护 x 轴，每个节点上存储在 mid 处最大的那条线段，于是每插入一条直线可以分类讨论是否与当前 mid 处的线段交换，而在 mid 处较小的那个线段一定在左右两侧中的一侧都不如 mid 处较大的线段。
- 于是我们每次只会向一侧递归，同时可以发现询问时只需要把路径上的每个线段都取个 max 即可，于是可以做到 $\mathrm{O}(n \log V)$。
- 当插入的不是直线而是线段时我们需要先分割为若干区间再单独插入，此时复杂度为 $\mathrm{O}(n \log^2 V)$。
- 最经典的应用就是斜率优化。

例题 · P4097 【模板】李超线段树

例题 · P4655 [CEOI2017] Building Bridges

容均摊线段树

- 对于一类信息在某些特殊情况下的维护往往很困难，我们能想到的办法只能递归解决。

容均摊线段树

- 对于一类信息在某些特殊情况下的维护往往很困难，我们能想到的办法只能递归解决。
- 我们可以选择维护信息的一些特征，称其为容，如果我们维护的过程会不断地使容减少，而容的总量我们又可以接受，那这时我们就可以接受暴力递归解决的复杂度。

容均摊线段树

- 对于一类信息在某些特殊情况下的维护往往很困难，我们能想到的办法只能递归解决。
- 我们可以选择维护信息的一些特征，称其为容，如果我们维护的过程会不断地使容减少，而容的总量我们又可以接受，那这时我们就可以接受暴力递归解决的复杂度。
- 直接说可能比较抽象，可以对照例题理解其思想。

例题 · P4145 上帝造题的七分钟 2

例题 · HDU5306 Gorgeous Sequence

例题 · LOJ 6507 「雅礼集训 2018 Day7」A

楼房重建线段树

- 又称单侧递归线段树，指的是一类以楼房重建为例的维护合并信息需要到子树内做查询操作的线段树。

楼房重建线段树

- 又称单侧递归线段树，指的是一类以楼房重建为例的维护合并信息需要到子树内做查询操作的线段树。
- 具体来说，这类线段树的信息是难以直接合并的，需要到子树的一侧去进行查询才能进行合并，同时查询的过程中只会向两侧中的一侧递归以保证复杂度。

楼房重建线段树

- 又称单侧递归线段树，指的是一类以楼房重建为例的维护合并信息需要到子树内做查询操作的线段树。
- 具体来说，这类线段树的信息是难以直接合并的，需要到子树的一侧去进行查询才能进行合并，同时查询的过程中只会向两侧中的一侧递归以保证复杂度。
- 显然总的复杂度应该是 $\mathrm{O}(n\log^2 n)$ 的，可以直接参考例题。

例题 · P4198 楼房重建

例题 · CF1340F Nastya and CBS

线段树历史和

- 即相比于传统线段树，新增了一个记录历史和的权值，以及一个记录历史值的操作：把当前权值向历史权值贡献一次。

线段树历史和

- 即相比于传统线段树，新增了一个记录历史和的权值，以及一个记录历史值的操作：把当前权值向历史权值贡献一次。
- 于是对于修改，我们需要同时维护传统的修改标记以及历史和标记，不失一般性地我们可以认为这两种标记是交替出现的，于是我们不妨用一个队列来维护原来的修改标记。

线段树历史和

- 即相比于传统线段树，新增了一个记录历史和的权值，以及一个记录历史值的操作：把当前权值向历史权值贡献一次。
- 于是对于修改，我们需要同时维护传统的修改标记以及历史和标记，不失一般性地我们可以认为这两种标记是交替出现的，于是我们不妨用一个队列来维护原来的修改标记。
- 然而直接暴力维护这个队列复杂度过于爆炸，注意到我们维护队列的目的是要完成标记对标记的合并以及标记对权值的合并，所以如果我们能总结出标记队列的某些特征，并只需要利用这些特征即可完成合并那么复杂度就降下来了。

线段树历史和

- 即相比于传统线段树，新增了一个记录历史和的权值，以及一个记录历史值的操作：把当前权值向历史权值贡献一次。
- 于是对于修改，我们需要同时维护传统的修改标记以及历史和标记，不失一般性地我们可以认为这两种标记是交替出现的，于是我们不妨用一个队列来维护原来的修改标记。
- 然而直接暴力维护这个队列复杂度过于爆炸，注意到我们维护队列的目的是要完成标记对标记的合并以及标记对权值的合并，所以如果我们能总结出标记队列的某些特征，并只需要利用这些特征即可完成合并那么复杂度就降下来了。
- 所以关键在于选取标记的特征，具体可以参考例题。

例题 · 忘了来源的题目 #2

给定一个序列，支持

- 区间加。
- 区间历史最大值。
- 区间历史和。

例题 · P4314 CPU 监控

更多例题

- CF932F Escape Through Leaf
- UOJ#228 基础数据结构练习题
- P4425 [HNOI/AHOI2018] 转盘
- CF1290E Cartesian Tree
- P3246 [HNOI2016] 序列
- CF997E Good Subsegments
- P8868 [NOIP2022] 比赛

例题 · 忘了来源的题目 #3

给定 n 个数求两两异或的最大值。

01 trie 的本质

- 本质上就是在区间 $[0, 2^k - 1]$ 上建立的一棵动态开点线段树。

01 trie 的本质

- 本质上就是在区间 $[0, 2^k - 1]$ 上建立的一棵动态开点线段树。
- 所以线段树合并，线段树的可持久化可以无差别直接套用到 01 trie 上。

例题 · P4735 最大异或和

例题 · P6623 [省选联考 2020 A 卷] 树

更多例题

- P5283 [十二省联考 2019] 异或粽子
- P4585 [FJOI2015] 火星商店问题

树的直径与重心

一般考虑不带权。直径的性质：

- 若直径长为奇数则所有直径的交为中间一条边；否则为中间一点。

重心的性质：

- 一棵树之多有两个重心，且它们相邻。
- 以树的重心为根时，所有子树的大小都不超过整棵树大小的一半。
- 树中所有点到某个点的距离和中，到重心的距离和是最小的。
- 把两棵树通过一条边相连得到一棵新的树，那么新的树的重心在连接原来两棵树的重心的路径上。

例题 · Atcoder [ABC221F] Diameter set

例题 · P1099 [NOIP2007 提高组] 树网的核

动态 DP

- 解决的是一类带修改的树形 dp 问题，首先树剖一下。

动态 DP

- 解决的是一类带修改的树形 dp 问题，首先树剖一下。
- 对于树上任意一节点，我们除了维护他子树的 dp 数组 f 外，我们还需要额外维护一个他子树中除了重儿子子树的其他点的 dp 数组 g。

动态 DP

- 解决的是一类带修改的树形 dp 问题，首先树剖一下。
- 对于树上任意一节点，我们除了维护他子树的 dp 数组 f 外，我们还需要额外维护一个他子树中除了重儿子子树的其他点的 dp 数组 g。
- 这时我们再看一条重链的转移就是线性的，于是我们如果修改一个点那么会影响他所在重链的 f，进而影响这条重链顶端父亲的 g，进而影响 f，以此类推。

动态 DP

- 解决的是一类带修改的树形 dp 问题，首先树剖一下。
- 对于树上任意一节点，我们除了维护他子树的 dp 数组 f 外，我们还需要额外维护一个他子树中除了重儿子子树的其他点的 dp 数组 g。
- 这时我们再看一条重链的转移就是线性的，于是我们如果修改一个点那么会影响他所在重链的 f，进而影响这条重链顶端父亲的 g，进而影响 f，以此类推。
- 结合树剖复杂度分析，我们只需要快速维护 f 的更新，g 的更新可以暴力，所以考虑将转移写成矩阵乘法的形式（一般是 $(\max, +)$），那么就能线段树快速维护转移。

例题 · P4719 【模板】”动态 DP”

例题 · P6021 洪水

DSU on tree

- 本质上是树剖 / 启发式合并在另一个角度的理解。

DSU on tree

- 本质上是树剖 / 启发式合并在另一个角度的理解。
- 即对于每个节点，我们如果只枚举非最大子树内的节点那么总复杂度是 $O(n \log n)$ 的。

DSU on tree

- 本质上是树剖 / 启发式合并在另一个角度的理解。
- 即对于每个节点，我们如果只枚举非最大子树内的节点那么总复杂度是 $\mathrm{O}(n \log n)$ 的。
- 而对于最大子树，我们可以考虑直接继承其信息。

例题 · CF600E Lomsat gelral

线段树合并

- 可以发现线段树合并与树这种结构十分匹配。

线段树合并

- 可以发现线段树合并与树这种结构十分匹配。
- 配合启发式合并等各类算法有很多玩法。

例题 · P4556 [Vani 有约会] 雨天的尾巴

例题 · P5298 [PKUWC2018]Minimax

虚树

- 给定一棵树，有若干次询问，每次询问会给出树上若干点问这些点之间的问题。

虚树

- 给定一棵树，有若干次询问，每次询问会给出树上若干点问这些点之间的问题。
- 我们可以将这些点与其 LCA 拿出来再建立一棵树，这样新树的大小是能接受的，我们也能在上面做一些正常的树上操作。

虚树

- 给定一棵树，有若干次询问，每次询问会给出树上若干点问这些点之间的问题。
- 我们可以将这些点与其 LCA 拿出来再建立一棵树，这样新树的大小是能接受的，我们也能在上面做一些正常的树上操作。
- 考虑如何建树，首先我们默认根一定在点集内，并将所有点按照 dfs 序排序。我们维护一个栈表示当前虚树根到当前点的路径上的所有点。

虚树

- 给定一棵树，有若干次询问，每次询问会给出树上若干点问这些点之间的问题。
- 我们可以将这些点与其 LCA 拿出来再建立一棵树，这样新树的大小是能接受的，我们也能在上面做一些正常的树上操作。
- 考虑如何建树，首先我们默认根一定在点集内，并将所有点按照 dfs 序排序。我们维护一个栈表示当前虚树根到当前点的路径上的所有点。
- 依次插入每个点 u，就不断弹栈直到栈顶是 u 的祖先则停止，将 u 入栈并与其连边，如果栈空那么将 u 与上一个点的 LCA 加入点集并连边。

虚树

- 给定一棵树，有若干次询问，每次询问会给出树上若干点问这些点之间的问题。
- 我们可以将这些点与其 LCA 拿出来再建立一棵树，这样新树的大小是能接受的，我们也能在上面做一些正常的树上操作。
- 考虑如何建树，首先我们默认根一定在点集内，并将所有点按照 dfs 序排序。我们维护一个栈表示当前虚树根到当前点的路径上的所有点。
- 依次插入每个点 u，就不断弹栈直到栈顶是 u 的祖先则停止，将 u 入栈并与其连边，如果栈空那么将 u 与上一个点的 LCA 加入点集并连边。
- 设 n 个点，虚树点集为 m，则复杂度为 $\mathrm{O}(n+m\log n)$，瓶颈在排序和求 LCA 上。

例题 · P3320 [SDOI2015] 寻宝游戏

例题 · P4103 [HEOI2014] 大工程

更多例题

- P3629 [APIO2010] 巡逻
- P4299 首都
- P5024 保卫王国
- P5327 [ZJOI2019] 语言
- P6773 [NOI2020] 命运
- P2495 [SDOI2011] 消耗战
- CF809E Surprise me!
- P5439 【XR-2】永恒

点分治

- 本质上是把序列分治那一套搬到树上，即每次找一个点为分治中心，考虑跨过中心的贡献，再对删除中心后的每个块分别计算。

点分治

- 本质上是把序列分治那一套搬到树上，即每次找一个点为分治中心，考虑跨过中心的贡献，再对删除中心后的每个块分别计算。
- 当我们选择树的重心为分治中心时，由其性质知每个子树大小至少减半，所以分治层数为 $O(\log)$ 层。

例题 · P4178 Tree

例题 · P5351 Ruri Loves Maschera

例题 · P4886 快递员

边分治

- 和点分治一样，不过把分治中心由点变为了边，可以选择两侧最大子树最小的那条边。（树为链时即序列分治）

边分治

- 和点分治一样，不过把分治中心由点变为了边，可以选择两侧最大子树最小的那条边。（树为链时即序列分治）
- 问题在于这样做不能保证复杂度（菊花图就寄了），于是我们考虑将树的形态转化。

边分治

- 和点分治一样，不过把分治中心由点变为了边，可以选择两侧最大子树最小的那条边。（树为链时即序列分治）
- 问题在于这样做不能保证复杂度（菊花图就寄了），于是我们考虑将树的形态转化。
- 如果给出的问题支持在树上增加虚点，那么我们可以通过增加虚点的方式使得整棵树的度数不超过 3，成为三度化。

边分治

- 和点分治一样，不过把分治中心由点变为了边，可以选择两侧最大子树最小的那条边。（树为链时即序列分治）
- 问题在于这样做不能保证复杂度（菊花图就寄了），于是我们考虑将树的形态转化。
- 如果给出的问题支持在树上增加虚点，那么我们可以通过增加虚点的方式使得整棵树的度数不超过 3，成为三度化。
- 边分治的好处是每次只会分成两部分，对于某些特殊问题会更容易处理一些。

例题 · P4220 [WC2018] 通道

点分树

- 用于点分治能解决的问题，但是多组询问且带修。

点分树

- 用于点分治能解决的问题，但是多组询问且带修。
- 我们将点分治的分治结构建树（以分治中心为代表），可以发现任意两个点的点分树上 LCA 在原树中一定在两点的简单路径上，且路径上其他点不会为公共祖先，且树高为 $\mathrm{O}(\log n)$。

点分树

- 用于点分治能解决的问题，但是多组询问且带修。
- 我们将点分治的分治结构建树（以分治中心为代表），可以发现任意两个点的点分树上 LCA 在原树中一定在两点的简单路径上，且路径上其他点不会为公共祖先，且树高为 $\mathrm{O}(\log n)$。
- 如果我们能将问题中对一个点的贡献分到其每个祖先上考虑，如路径问题，对一个固定 x 以及任意从 y 的路径统计，我们可以枚举 x 在点分树上的祖先进行计算，于是复杂度便与树高相关。

例题 · P6329 【模板】点分树

更多例题

- P3060 [USACO12NOV] 平衡树 Balanced Trees
- P5306 [COCI2019] Transport
- P4565 [CTSC2018] 暴力写挂
- P3241 [HNOI2015] 开店
- P3345 [ZJOI2015] 幻想乡战略游戏

根号分治

- 一种隐式分块，即设置一个阈值，把问题中中大于某个阈值的称为大块，其余称为小块。

根号分治

- 一种隐式分块，即设置一个阈值，把问题中中大于某个阈值的称为大块，其余称为小块。
- 可以发现大块的数量不会很多，而小块的值域很小，所以对它们分别讨论不同的解决方法。

例题 · P3396 哈希冲突

例题 · HDU 6115 Factory

例题 · CF348C Subset Sums

序列分块

- 即设置一个阈值 B 后将序列每 B 个分成一块，如果我们能维护好整块和散块之间的关系那么可能能够解决问题。

序列分块

- 即设置一个阈值 B 后将序列每 B 个分成一块，如果我们能维护好整块和散块之间的关系那么可能能够解决问题。
- 相比传统数据结构能维护一些更难维护的东西，同样很灵活。

例题 · 分块四问

- 区间修改 $O(\sqrt{n})$ 单点查询 $O(1)$

例题 · 分块四问

- 区间修改 $O(\sqrt{n})$ 单点查询 $O(1)$
- 单点修改 $O(1)$ 区间查询 $O(\sqrt{n})$

例题 · 分块四问

- 区间修改 $O(\sqrt{n})$ 单点查询 $O(1)$
- 单点修改 $O(1)$ 区间查询 $O(\sqrt{n})$
- 区间修改 $O(\sqrt{n})$ 区间查询 $O(1)$

例题 · 分块四问

- 区间修改 $O(\sqrt{n})$ 单点查询 $O(1)$
- 单点修改 $O(1)$ 区间查询 $O(\sqrt{n})$
- 区间修改 $O(\sqrt{n})$ 区间查询 $O(1)$
- 区间修改 $O(1)$ 区间查询 $O(\sqrt{n})$

例题 · P2801 教主的魔法

例题 · P4168 [Violet] 蒲公英

更多例题

- CF1654E Arithmetic Operations
- CF1039D You Are Given a Tree
- P5356 [Ynoi2017] 由乃打扑克
- P3203 [HNOI2010] 弹飞绵羊
- P4135 作诗
- CodeChef Chef and Churu
- P4117 [Ynoi2018] 五彩斑斓的世界
- P5397 [Ynoi2018] 天降之物

例题 · P5268 [SNOI2017] 一个简单的询问

例题 · P4396 [AHOI2013] 作业

带修莫队

- 我们把时间轴和左端点右端点一样看成一个需要移动的指针。

带修莫队

- 我们把时间轴和左端点右端点一样看成一个需要移动的指针。
- 于是如果每次移动一个单位的时间指针是容易的话我们就能把问题看成一个三维莫队。

带修莫队

- 我们把时间轴和左端点右端点一样看成一个需要移动的指针。
- 于是如果每次移动一个单位的时间指针是容易的话我们就能把问题看成一个三维莫队。
- 类比莫队的时间复杂度分析可知所有指针移动代价都为 $O(1)$ 时复杂度为 $O(nq^{2/3})$。

例题 · P1903 [国家集训队] 数颜色

回滚莫队

- 用于解决加入一个元素容易，但是删除一个元素困难的问题（仍然时改删除为撤销）

回滚莫队

- 用于解决加入一个元素容易，但是删除一个元素困难的问题（仍然时改删除为撤销）
- 我们可以把莫队的时间复杂度分析具体化下来，即枚举左端点所在的块，然后将右端点整体向右扫。

回滚莫队

- 用于解决加入一个元素容易，但是删除一个元素困难的问题（仍然时改删除为撤销）
- 我们可以把莫队的时间复杂度分析具体化下来，即枚举左端点所在的块，然后将右端点整体向右扫。
- 位于块内的元素先不加入，当右端点扫到一个询问的右端点时再加入左块的元素，询问解决后再撤销。

回滚莫队

- 用于解决加入一个元素容易，但是删除一个元素困难的问题（仍然时改删除为撤销）
- 我们可以把莫队的时间复杂度分析具体化下来，即枚举左端点所在的块，然后将右端点整体向右扫。
- 位于块内的元素先不加入，当右端点扫到一个询问的右端点时再加入左块的元素，询问解决后再撤销。
- 可以发现复杂度并没有变化，但是我们可以不用再去实现删除操作。

例题 · P5906 【模板】回滚莫队

例题 · P4137 Rmq Problem

莫队二次离线

- 有一类问题如果我们使用朴素莫队时移动指针需要的信息需要比较复杂的计算才能得到。

莫队二次离线

- 有一类问题如果我们使用朴素莫队时移动指针需要的信息需要比较复杂的计算才能得到。
- 而如果我们将莫队每次移动指针需要的信息当作一次询问，那么显然我们是可以把这些询问再离线下来的。

莫队二次离线

- 有一类问题如果我们使用朴素莫队时移动指针需要的信息需要比较复杂的计算才能得到。
- 而如果我们将莫队每次移动指针需要的信息当作一次询问，那么显然我们是可以把这些询问再离线下来的。
- 于是我们可以尝试用一些算法去离线解决这部分计算，同时注意空间上的消耗。

例题 · P5047 [Ynoi2019 模拟赛] Yuno loves sqrt technology II

更多例题

- P4462 [CQOI2018] 异或序列
- P1494 [国家集训队] 小 Z 的袜子
- P4689 [Ynoi2016] 这是我自己的发明
- P4688 [Ynoi2016] 掉进兔子洞
- P8078 [WC2022] 秃子酋长
- CF940F Machine Learning
- P5386 [Cnoi2019] 数字游戏
- P5501 [LnOI2019] 来者不拒，去者不追